Вопросы к экзамену по дисциплине "Основы информатики"

1. Структура информатики. Понятие информации, ее измерение и представление.

Информатика – это комплексная техническая наука, которая систематизирует приёмы создания, сохранения, воспроизведения, обработки и передачи данных средствами вычислительной техники, а также принципы функционирования этих средств и методы управления ними

Информация – это совокупность сведений (данных), которая воспринимается из окружающей среды(входная информация), выдаётся в окружающую среду(исходная информация) или сохраняется внутри определённой системы. Информация существует в виде документов, чертежей, рисунков, текстов, звуковых и световых сигналов, электрических и нервных импульсов и т.п. Информация измеряется в битах(наименьшая единица) и в байтах (1 байт = 8 бит)

2. Сигнал, сообщение, алфавит, основание кода. Единицы количества информации.

Код – система условных знаков (символов) для передачи, обработки и хранения информации (сообщения)

Кодирование – процесс представления информации (сообщения) в виде кода

Алфавит кодирования – множество символов, используемых для кодирования. Например в памяти компьютера любая информация кодируется с помощью двоичного алфавита содержащего всего два символа: 0 и 1

3. Позиционные и непозиционные системы исчисления. Алгоритмы перевода чисел из одной позиционной системы исчисления в другую

В непозиционных системах счисления значение цифры не зависит от положения в числе. В позиционных системах счисления величина, обозначаемая цифрой в записи числа, зависит от её позиции. Современная математика использует позиционную систему счисления, её основание равно 10, запись любых чисел производится с помощью десяти цифр от 0 до 9

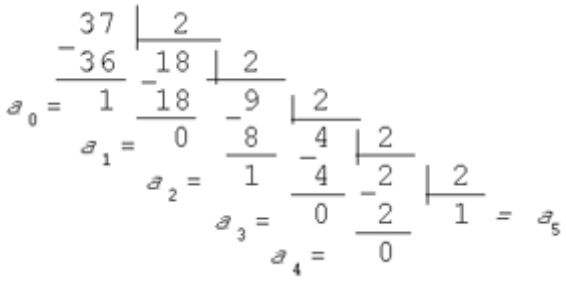
При переводе чисел из десятичной системы счисления в систему с основанием $P > 1$ обычно используют следующий алгоритм:

1) если переводится целая часть числа, то она делится на P, после чего запоминается остаток от деления. Полученное частное вновь делится на P, остаток запоминается. Процедура продолжается до тех пор, пока частное не станет равным нулю. Остатки от деления на P выписываются в порядке, обратном их получению;

2) если переводится дробная часть числа, то она умножается на P, после чего целая часть запоминается и отбрасывается. Вновь полученная дробная часть умножается на P и т.д. Процедура продолжается до тех пор, пока дробная часть не станет равной нулю. Целые части выписываются после двоичной запятой в порядке их получения. Результатом может быть либо конечная, либо периодическая двоичная дробь. Поэтому, когда дробь является периодической, приходится обрывать умножение на каком-либо шаге и довольствоваться приближенной записью исходного числа в системе с основанием P.

Пример 1. Перевести число 37_{10} в двоичную систему.

Для обозначения цифр в записи числа используем символику: $a_5a_4a_3a_2a_1a_0$



Отсюда: 37_{10} = l00l0l$_2$

4. Представление данных в памяти компьютера. Прямой, обратный, дополнительный код.

Для представления любой информации в памяти ЭВМ (как числовой, так и не числовой) используется двоичный способ кодирования.

Элементарная ячейка памяти ЭВМ имеет длину 8 бит (1 байт). Каждый байт имеет свой номер (его называют **адресом**). Наибольшую последовательность бит, которую ЭВМ может обрабатывать как единое целое, называют **машинным словом**. Длина машинного слова зависит от разрядности процессора и может быть равной 16, 32, 64, 128 и т. д. битам.

Разрядной сеткой компьютера называется совокупность запоминающих элементов для размещения данных. **Форматом** называется способ размещения данных в разрядной сетке.

Прямой код – способ представления двоичных чисел с фиксированной запятой в компьютерной арифметике. Главным образом используется для записи положительных чисел. Прямой код записывается в виде двоичного числа и слева дополняется 0 до выбранной разрядности. Если запись со знаком

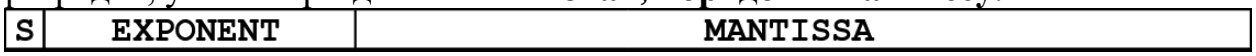
Для представления чисел в форматах со знаком используется **дополнительный код**. Использование дополнительного кода позволяет заменить операцию вычитания числа операцией сложения с дополнительным кодом этого числа.

Для представления целого числа в дополнительном коде используется следующий алгоритм:

- число переводится в двоичную систему;
- результат дополняется нулями слева в пределах выбранного формата;
- полученное число переводится в **обратный код.** Обратный код для положительного двоичного числа совпадает с его прямым кодом, а для отрицательного числа нужно во всех разрядах, кроме знакового, нули заменить единицами, а единицы нулями

5. Представление действительных чисел с плавающей точкой

Вещественные числа обычно представляются в форматах с плавающей запятой. Формат с плавающей запятой состоит из набора отдельных двоичных разрядов, условно разделенных на **знак**, **порядок** и **мантиссу**.



Формат double (число двойной точности) — компьютерный формат представления числа с плавающей запятой, занимающий в памяти 64 бита, или

8 байт. В формате двойной точности double для хранения знака отводится 1 бит, для хранения порядка – 11 бит, а для хранения мантиссы – 52 бита. Знаковый разряд положительных чисел всегда равен 0, отрицательных – 1.

В форматах чисел с плавающей запятой хранятся смещенные порядки Псм=П+ΔП. Для формата с плавающей запятой двойной точности double смещение ΔП = 1023. Представление порядка со смещением позволяет упростить операции сравнения чисел и отказаться от использования знакового разряда порядка, т.к. смещенный порядок всегда положительный.

Значение числа с плавающей запятой определяется по формуле:

$$C = (-1)^S 2^{П_{СМ} - \Delta П} (F_0, F_1, \ldots, F_i, \ldots, F_n),$$

где S – знак числа, F – разряды числа, причем F_0 всегда равен 1, поэтому в памяти компьютера не хранится.

Для перевода числа в формат с плавающей запятой используется следующий алгоритм:

- число переводится в двоичную систему;

- мантисса числа в двоичной системе счисления преобразуется к нормализованному виду, для этого запятая сдвигается на место после первой единицы в числе, каждый сдвиг запятой влево эквивалентен делению на 2, поэтому должен быть компенсирован умножением числа на 2, и наоборот каждый сдвиг запятой вправо эквивалентен умножению на 2, поэтому должен быть компенсирован делением числа на 2;

- определяется и помещается в разрядную сетку знак числа (знак 0 для положительных чисел и 1 для отрицательных);

- определяется смещенный порядок числа, для этого к количеству знаков, на которое сместилась запятая при переводе мантиссы к нормализованному виду добавляется смещение (ΔП = 1023 для типа double), далее смещенный порядок переводится в двоичную систему счисления и помещается в разрядную сетку;

- дробная часть мантиссы помещается в разрядную сетку, при необходимости мантисса дополняется нулями справа в пределах выбранного формата;

- результат переводится в шестнадцатеричную систему счисления.

6. Базовые элементы языка Си.

К базовым элементам языка Си относится:

Вывод, типы данных, операции, ввод, выполнение по условию, выполнение в цикле, подпрограммы

7. Функции ввода-вывода на языке Си. Форматы ввода-вывода.

Вывод:

printf () – форматированный вывод

puts () – вывод строки с автоматическим переводом строки(автоматически добавляет \n

putchar () – вывод одного символа

Форматы вывода:

%d – целое число (int)

%u – беззнаковое целое (unsigned int)

%ld, %li – длинное целое (long int)

%lu – беззнаковое длинное целое (unsigned long)

%f – вещественное число (float/double)

%lf – вещественное число (double)

%c -символ (char)

%s – строка (char *)

%p – указатель (void *)

%% - символ %

Ввод:

scanf () – форматированный ввод

gets () или fgets () – чтение строки

getchar () – чтение одного символа

Форматы ввода:

%d – целое число (int)

%u – беззнаковое целое (unsigned int)

%ld – длинное целое (long int)

%lu – беззнаковое длинное целое (unsigned long)

%f – вещественное число (float/double)

%lf – вещественное число (double)

%c -символ (char)

%s – строка (до первого пробела)

%p – указатель (void *)

8. Логические (булевские) операторы. Основные эквивалентности для булевых функций.

9. Операции в языке Си.

= - присвоение

a= - b – унарный минус

a= + b – унарный плюс

! – логическое НЕ

a=-b – поразрядное дополнение

a=&b – адрес

a=*intptr – указатель(ссылка)

a=sizeof(b) – размер

a++ и ++a – увеличение на 1

a-- и --a – уменьшение на 1

a= b*c – умножение

a= b/c – целочисленное деление

x= b/c – деление

a= b%c – модуль(остаток)

a=b+c – сложение

a=b-c – вычитание

a=b>>c – сдвиг вправо

a=b<<c – сдвиг влево

a > b – больше, чем

a >= b – больше или равно, чем

a < b – меньше, чем

a <=b – меньше или равно, чем

a == b – равно

a != b – не равно

a=b&c – поразрядное И

a=b|c – поразрядно ИЛИ

a=b^c – поразрядное исключающее ИЛИ

flag1 && flag2 – логическое И

flag1 || flag2 – логическое ИЛИ

a=b - присвоение

10. Функции в языке Си. Рекурсивные функции. Прототипы функций.

Функция состоит из:

1. Типа возвращаемого значения (или void, если ничего не возвращает)
2. Имени функции
3. Параметров (в скобках, могут отсутствовать)
4. Тела функции (в фигурных {} скобках)

Прототип функции сообщает компилятору о её существовании до фактического определения. Это нужно, если функция вызывается до своего объявления. Нужны чтобы вызывать функцию до её определения, помогают избежать ошибок компиляции, улучшают читаемость кода.

Рекурсивные функции – это когда функция вызывает саму себя. Рекурсия должна иметь условие выхода, иначе произойдёт переполнение стека.

Пример:

```
int factorial(int  ) {
    if (  <= 1) { // Условие выхода
        return 1;
    }
    return   * factorial(  - 1); // Рекурсивный вызов
}
```

11. Указатели в языке Си. Динамическое распределение памяти. Массивы указателей в языке Си

Указатель – это переменная, которая хранит адрес другой переменной в памяти. Основные операции с указателями: & - взятие адреса переменной, * - разыменование (доступ к значению по адресу)

Динамическое выделение памяти необходимо для эффективного использования памяти компьютера. Основано на операции sizeof которая определяет размер участка памяти в байтах, который компилятор отводит под массив после его объявления.

Основные функции для работы с динамической памятью:

calloc () – выделение + обнуление памяти

malloc () – выделение памяти

realloc () – изменение размера выделенной памяти

free () – освобождение памяти

В языке *C* можно использовать массивы указателей, элементы которых содержат как правило указатели на строковые данные. Объявляется такой массив следующим образом:

*char *m[5];*

Здесь массив *m[5]* – массив, который может содержать пять адресов данных типа *char*.

Массив указателей можно при объявлении инициализировать, т.е. назначать при объявлении его элементам конкретные адреса заданных строк

12. Массивы и строки в языке Си. Функции для работы со строками.

В Си строка – это массив символов типа char заканчивающийся нуль-терминатором \0

Для работы со строками используются функции из библиотеки string.h:

strcpy и strncpy – копирование строк

strcat и strncat – объединение строк

strncmp – сравнение строк, 0 – если строки равны, <0 если s1<s2, >0 если s1>s2

strlen – длина строки

strchr – поиск первого вхождения символа в строке

strstr – поиск первого вхождения подстроки(слова) в строке

13. Массивы и функции в языке Си.

14. Структуры и объединения

15. Функции ввода/вывода

Вывод:

printf () – форматированный вывод

puts () – вывод строки с автоматическим переводом строки(автоматически добавляет \n

putchar () – вывод одного символа

Ввод:

scanf () – форматированный ввод

gets () или fgets () – чтение строки

getchar () – чтение одного символа

16. Поколения ЭВМ

17. Основные функциональные элементы ЭВМ. Триггер. Дешифратор. Шифратор. Счетчик.

18. Основные функциональные элементы ЭВМ. Регистры. Арифметико-логическое устройство. Запоминающие устройства. Устройства управления.

19. Архитектура фон Неймана

20. Гарвардская архитектура

21. Обзор системы Windows. Архитектура системы Windows.

22. Основные принципы построения ОС

23. Основные идеи, заложенные в операционную систему MS Windows.

24. Процессы и потоки в Windows. Объекты ядра Windows.

25. Распределение времени между потоками в Windows. Классы приоритетов процессов.

26. Файловые системы.

27. Языки программирования

28. Базы данных и системы управления базами данных. Классификация баз данных

29. Этапы и элементы процесса разработки

30. Классификация вычислительных сетей. Локальные вычислительные сети. Глобальные сети. Интернет.

31. Протоколы передачи данных

32. Средства автоматизации инженерных и научных расчетов